|  |  |  |
| --- | --- | --- |
| **Nama & NPM** | **Topik:** | **Tanggal:** |
| **AriIq Ashar Sofyan**  **G1F024052** | **IF & SWITCH** | **10 Oktober 2024** |

**[No.1] Identifikasi Masalah:**

**Contoh 1:**

**import java.util.Scanner; //memanggil impor package yang membaca masukan pengguna**

**public class PercabanganIf {  
    public static void main(String[] args) {  
        Scanner input = new Scanner(System.in);  // membaca teks yang dimasukkan pengguna  
        System.out.print("Masukkan Angka Anda : ");  //pengguna memasukkan data  
        nilai =  masuk.nextByte();  //menyimpan masukan pengguna ke tipe data  
          
        if (nilai = 1000) {  //percabangan yang memeriksa kondisi  
        System.out.println("Seribu");  //baris kode yang dieksekusi bila benar  
                }  
        else  {  //baris kode yang dieksekusi bila kondisi tidak terpenuhi dan salah  
            System.out.println("Nilai Bukan Seribu");  
        }    }  
}**

**contoh 2:**

**import java.util.Scanner;**

**public class IfBersarang {  
    public static void main(String[] args) {  
        Scanner varT = new Scanner(System.in);  
        System.out.print("Masukkan Angka Tugas Anda : ");  
        int nilaiT =  varT.nextByte();  
      
        Scanner varQ = new Scanner(System.in);  
        System.out.print("Masukkan Angka Quiz Anda : ");  
        int nilaiQ =  varQ.nextByte();  
                 
    if (nilaiU >= 80) {  
        if(nilaiT >= 80) {  
            System.out.println("Anda mendapatkan nilai A");  
        }  
    }  
    else{  
        System.out.println("Anda TIDAK mendapatkan nilai A");  
    }  
    }  
}**

**[No.1] Analisis dan Argumentasi**

1. Saya mengusulkan permasalahan ini :

1.1. Evaluasi penyebab kesalahan dan perbaiki kode tersebut!

1.2. Cermati contoh 2, analisa kondisi pada IF bersarang! Tambahkan satu kondisi IF dengan satu nilai input Quiz (nilaiQ). Jika nilai UTS, Tugas, dan Quiz lebih besar sama dengan 80 maka siswa mendapat nilai A.

1.3. Apakah ketiga kondisi IF pada Contoh 1.2. dapat diringkas menjadi satu kondisi? Periksa satu kondisi mana yang paling tepat menggantikan ketiga kondisi itu!

a. IF (nilaiU >= 80 || nilaiT >= 80 || nilaiQ >= 80)

b. IF (nilaiU >= 80 || nilaiT >= 80 && nilaiQ >= 80)

c. IF (nilaiU >= 80 && nilaiT >= 80 || nilaiQ >= 80)

d. IF (nilaiU >= 80 && nilaiT >= 80 && nilaiQ >= 80)

1.4. Uraikan gambar diagram flowchart dari Latihan 1.2!

Jawaban:

* 1. Contoh 1:
* Variabel nilai belum dideklarasikan sebagai byte.
* Nama "masuk.byte" tidak cocok dengan nama di input scanner.
* Operator perbandingan harus menggunakan dua tanda sama dengan (==) bukan satu (=) yang berarti penugasan.
* Nextbyte memiliki batas nilai tertentu.

contoh 2:

* variable nilaiU yang tidak cocok
* belum ada operator untuk nilai UTS
  1. Kondisi IF bersarang ini memeriksa satu per satu apakah nilai UTS, Tugas, dan Quiz semuanya ≥ 80. Jika semua terpenuhi, siswa mendapatkan nilai A.
  2. d. if (nilaiU >= 80 && nilaiT >= 80 && nilaiQ >= 80)

Kondisi ini memeriksa apakah ketiga nilai (UTS, Tugas, dan Quiz) harus ≥ 80 untuk mendapatkan nilai A. Kondisi lain hanya memerlukan satu atau dua nilai yang memenuhi syarat, padahal semua tiga nilai harus memenuhi syarat.

* 1. Flowchard akan ditampilkan dengan gambar

**[No.1] Algoritma dan Kode Program**

1. Algoritma

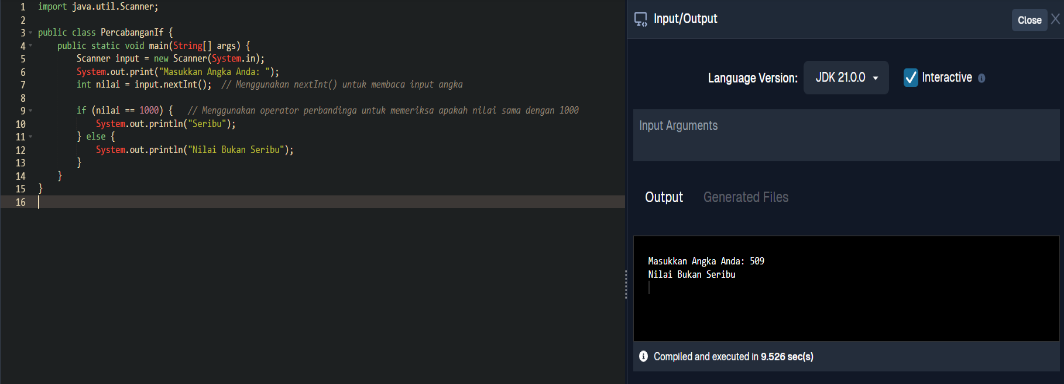
Contoh 1:

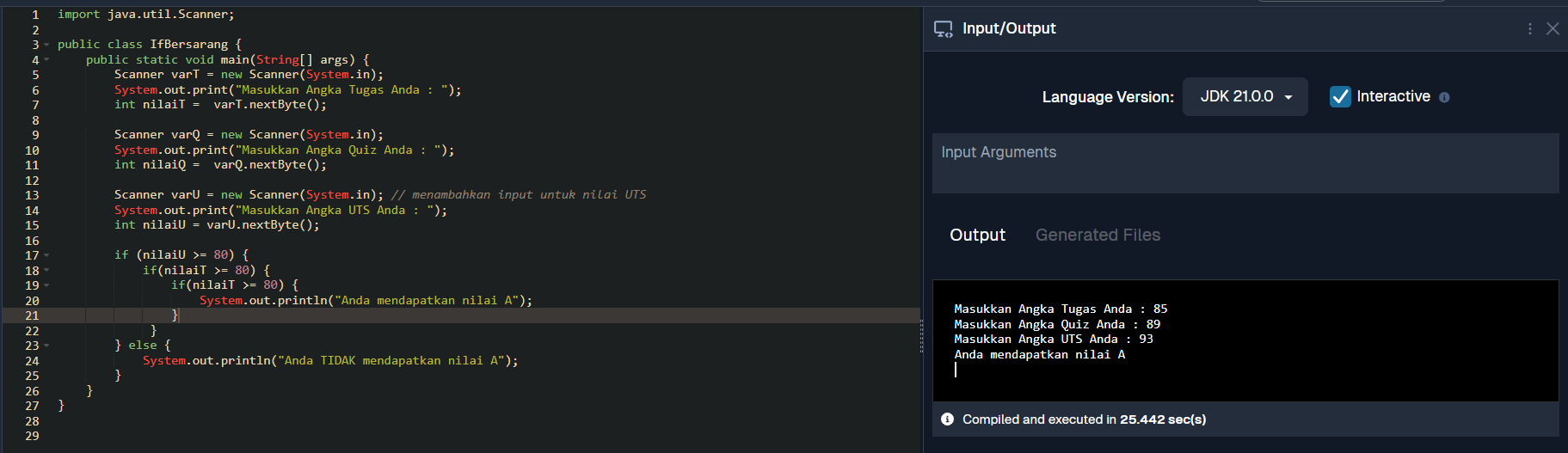
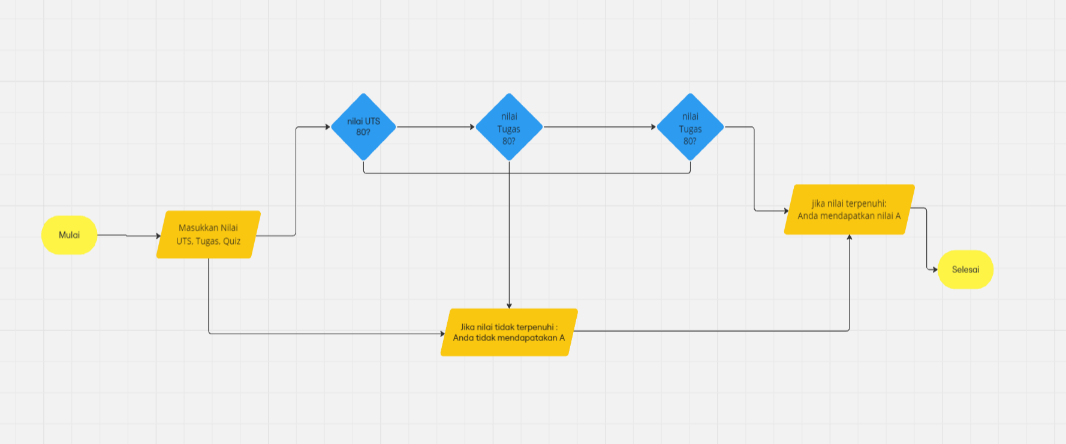
* Mulai program.
* Import java.util.Scanner.
* Buat kelas publik bernama percabangan.
* Buat metode utama main(String[] args) di dalam kelas percabangan.
* Buat objek Scanner dengan new Scanner(System.in) di dalam metode main.
* Tampilkan output menggunakan System.out.println di dalam metode main.
* Gunakan pernyataan if-else di dalam metode main.
* Program selesai.

Contoh 2:

* Mulai program
* Import java.util.Scanner.
* Buat kelas publik bernama ifBersarang.
* Buat metode utama main(String[] args) di dalam kelas ifBersarang.
* Buat objek Scanner dengan new Scanner(System.in) di dalam metode main.
* Tampilkan output dengan System.out.println() di dalam metode main.
* Deklarasikan variabel int nilai di dalam metode main.
* Gunakan pernyataan if-else di dalam metode main.
* Program selesai.

1. Kode program contoh 1:



1. Kode program contoh 2:  
   
2. Flowchard  
   

**[No.1] Kesimpulan**

Pada program itu Saya menggunakan kelas public karena memungkinkan program diakses dari luar dan dieksekusi langsung oleh compiler. Kelas public juga standar untuk program utama di Java. Perbaikan dilakukan dengan menambahkan deklarasi variabel dan operator perbandingan yang benar, karena Java mengharuskan variabel dideklarasikan sebelum digunakan, serta menggunakan operator perbandingan (==) untuk memeriksa kesamaan, bukan operator penugasan (=).

**[No.2] Identifikasi Masalah:**

**Contoh 1:**

**import java.util.Scanner;**

**public class SwitchBersarang {  
    public static void main(String[] args) {  
        Scanner masukData = new Scanner(System.in);  
        // mengambil input  
        System.out.print("Pilih A atau B :  ");  
        char data = data.next().charAt(0);   
        switch(data):    
        case A   
            System.out.print("Anda sudah rajin belajar");  
            break; // baris 1  
        case 'B':   
            System.out.print("  Anda perlu kurangi main game");  
            break; // baris 2  
        default  
            System.out.print("  Pilihan anda diluar A atau B ");   
            break;  
}    }**

**contoh 2:**

**import java.util.Scanner;**

**public class SwitchBersarang {  
    public static void main(String[] args) {  
         byte bulan;  
            int tahun = 2022;  
            int jumlahHari = 0;  
            System.out.print("Masukkan data bulan (dalam angka): ");  
            Scanner masukData = new Scanner(System.in);  
            bulan = masukData.nextByte();  
              
            switch (bulan) {  
                    case 1: jumlahHari = 31; break;  
            case 2: if (tahun % 4 == 0) {  jumlahHari = 29; }  
                    else { jumlahHari = 28; }  
                    break;  
case 3: jumlahHari = 31; break;  
case 4: jumlahHari = 30; break;  
case 5: jumlahHari = 31; break;  
case 6: jumlahHari = 30; break;  
case 7: jumlahHari = 31; break;  
case 8: jumlahHari = 31; break;  
case 9: jumlahHari = 30; break;  
case 10: jumlahHari = 31; break;  
case 11: jumlahHari = 30; break;  
case 12: jumlahHari = 31; break;  
                    default: System.out.println("Maaf bulan hanya sampai 12.");  
                    break;  
            }  
            System.out.println("Jumlah hari = " + jumlahHari);  
}    }**

**[No.2] Analisis dan Argumentasi**

2.1. Perhatikan kode pada Contoh 3. Evaluasi kesalahan dan perbaiki kode tersebut! Hapus kode break; di //baris 1, lalu jalankan. Kemudian, hapus break; di //baris 2, dan jalankan lagi. Simpulkan fungsi break pada switch!

2.2. Periksa kode pada Contoh 4. Apakah sudah efisien? Apakah ada informasi yang diulang? Jika ya, berikan saran penulisan yang lebih tepat!

2.3. Apakah masalah pada Contoh 3 bisa diubah menjadi perintah IF? Jika bisa, berikan bentuk perintah IF dari Contoh 3! Bandingkan cara IF dan SWITCH menyelesaikan masalah.

2.4. Gambar flowchart untuk Latihan 2.2 dan Latihan 2.3!

Jawaban:

2.1. Penyebab Kesalahan**:**

1. Baris char data = data.next().charAt(0); salah karena variabel data belum dideklarasikan. Harusnya variabelnya masukData.
2. Tanda titik dua (:) pada switch(data): salah, harusnya pakai kurung kurawal { untuk memulai blok switch.
3. case 'A' harus menggunakan tanda kutip tunggal ('A') karena 'A' adalah karakter, bukan variabel.
4. Pada default, tidak ada kurung kurawal { atau :, sehingga terjadi kesalahan sintaks.

Fungsi break: Jika break dihapus, eksekusi akan berlanjut ke case berikutnya tanpa melihat nilai switch, disebut "fall-through". break diperlukan untuk menghentikan eksekusi setelah setiap case.

2.2. Efisiensi: Penulisan jumlah hari untuk setiap bulan berulang-ulang. Ini bisa dioptimalkan dengan mengelompokkan bulan yang jumlah harinya sama dalam satu case. Dengan menggabungkan beberapa case, kodenya jadi lebih ringkas dan mudah dipahami.

2.3. Perbandingan IF dan SWITCH**:**

* Switch lebih efisien dan mudah dibaca jika ada banyak pilihan tetap, seperti pilihan A atau B.
* If lebih fleksibel karena bisa menangani ekspresi yang lebih kompleks, tapi bisa kurang efisien jika ada banyak pilihan tetap.

2.4. flowchard ditampilkan dengan gambar

**[No.2] Algoritma dan Kode Program**

1. Algoritma

Contoh 1:

* Mulai program.
* Import java.util.Scanner.
* Buat kelas publik switchBersarang.
* Buat metode utama main(String[] args) di dalam kelas switchBersarang.
* Buat objek Scanner dengan new Scanner(System.in) di dalam metode main.
* Tampilkan output dengan System.out.println() di dalam metode main.
* Deklarasikan variabel char dan blok switch di dalam metode main.
* Gunakan case dan break di dalam switch.
* Program selesai.

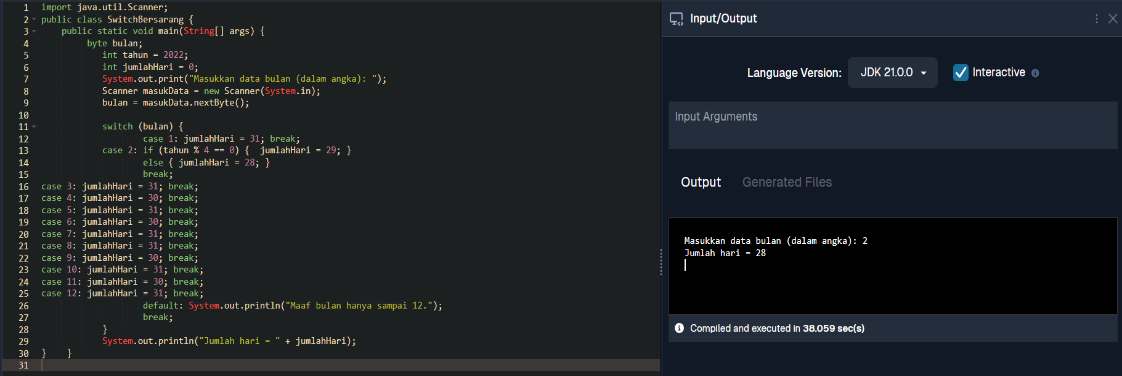
Contoh 2:

* Mulai program.
* Import java.util.Scanner.
* Buat kelas publik switchBersarang.
* Buat metode utama main(String[] args) di dalam kelas switchBersarang.
* Deklarasikan variabel byte dan int di dalam metode main.
* Buat objek Scanner dengan new Scanner(System.in) di dalam metode main.
* Tampilkan output dengan System.out.println() di dalam metode main.
* Gunakan case dan break di dalam blok switch.
* Program selesai.

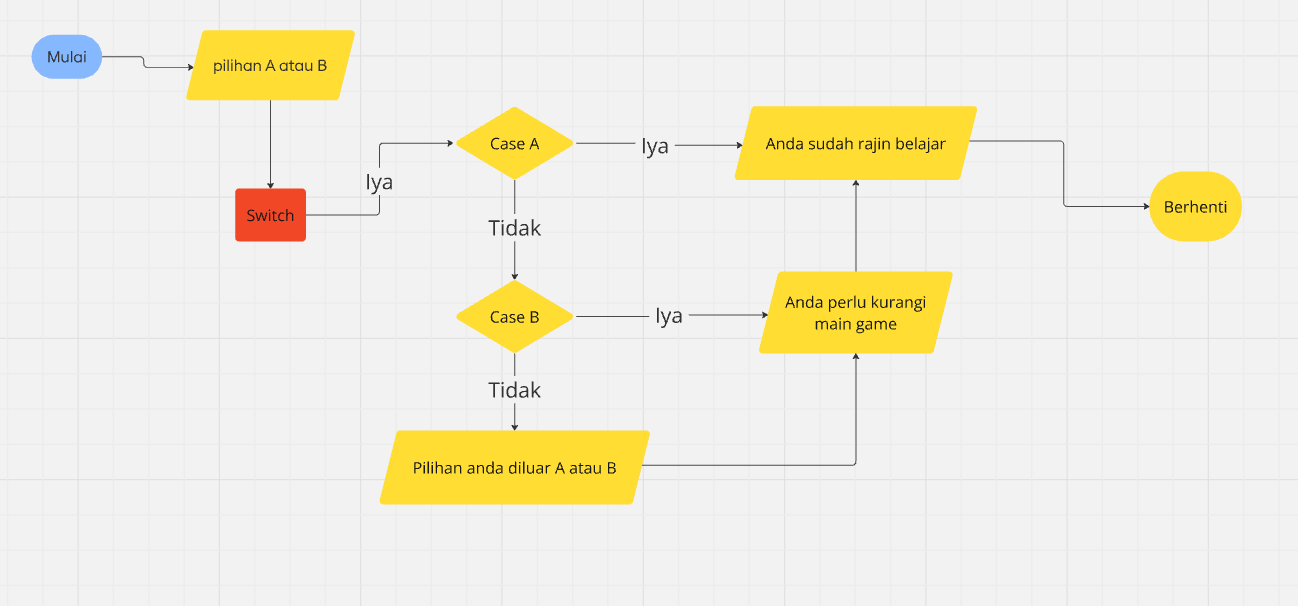
1. Kode program contoh 1:



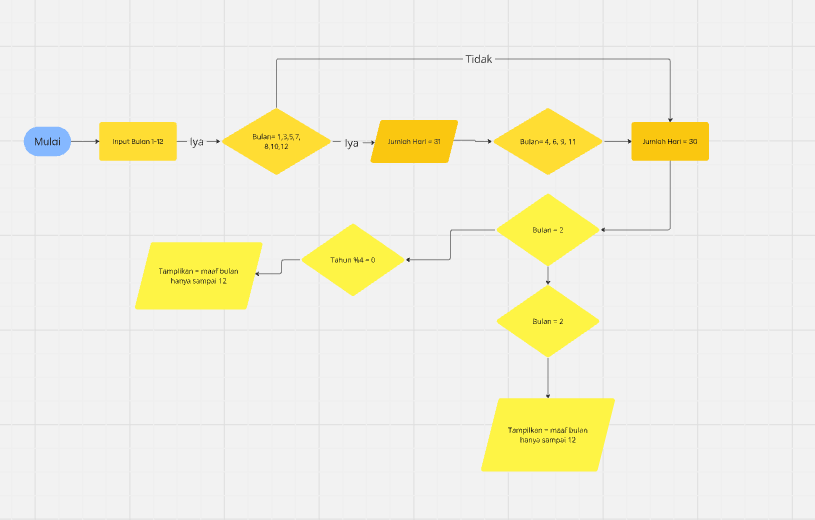
1. Kode program contoh 2:



1. Flowchard 2.1



1. Flowchard 2.2



**[No.2] Kesimpulan**

Saya menggunakan kelas public karena perlu diakses dari luar file atau package, terutama untuk menjalankan metode utama (main). Perbaikan program dilakukan dengan menambahkan kurung kurawal {} dan menggunakan tanda kutip tunggal (' ') untuk karakter case, karena Java mengharuskan kurung kurawal untuk blok kode dan karakter harus ditulis dengan tanda kutip tunggal agar dibedakan dari variabel.